

**Technical Report
1017**

Machine Learning for a Toolkit for Image Mining

**R.L. Delanoy
R.J. Sasiela**

6 March 1995

Lincoln Laboratory

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

LEXINGTON, MASSACHUSETTS



Prepared for the Department of the Air Force under Contract F19628-95-C-0002.

Approved for public release; distribution is unlimited.

ADA293137

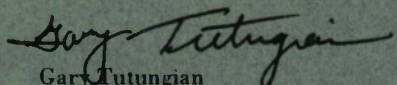
This report is based on studies performed at Lincoln Laboratory, a center for research operated by Massachusetts Institute of Technology. The work was sponsored by the Department of the Air Force under Contract F19628-95-C-0002.

This report may be reproduced to satisfy needs of U.S. Government agencies.

The ESC Public Affairs Office has reviewed this report, and it is releasable to the National Technical Information Service, where it will be available to the general public, including foreign nationals.

This technical report has been reviewed and is approved for publication.

FOR THE COMMANDER


Gary Tutungian
Administrative Contracting Officer
Contracted Support Management

Non-Lincoln Recipients

PLEASE DO NOT RETURN

Permission is given to destroy this document
when it is no longer needed.

MASSACHUSETTS INSTITUTE OF TECHNOLOGY
LINCOLN LABORATORY

**MACHINE LEARNING FOR A
TOOLKIT FOR IMAGE MINING**

R.L. DELANOY
Group 21

R.J. SASIELA
Group 42

TECHNICAL REPORT 1017

6 MARCH 1995

Approved for public release; distribution is unlimited.

ABSTRACT

Given the exponential growth in the number of images stored in databases around the world, there is a clear need for a computer-assisted means of searching images. Key words and computed indices, referred to as "metadata," can be used to tag images for subsequent retrieval. But this approach is sufficient for only a few applications. Most potential applications will require searching for images on the basis of image content. However, existing techniques for content-based search are either too slow, customized at great expense for a single application, or simply do not work very well.

An environment has been created enabling a user with very limited computer skills to interactively train a computer algorithm to recognize patterns of spectral or textural features in imagery. The trained algorithm can then be exported as an independent agent to search large databases for matching patterns. Following retrieval of images having matching patterns, a user can further refine the agent's performance by indicating mistakes. The final product is a search tool capable of prescreening and highlighting images, significantly reducing the work load of analysts attempting to detect regions or objects in imagery.

An approach to interactive learning has been developed as part of this environment. Based on techniques of knowledge-based image processing, this approach uses *interest images* to provide a means of continuous feedback of algorithm performance to the user. The user in turn responds by indicating where the algorithm is making mistakes. The set of user-indicated examples and counterexamples form the inputs to a new machine learning algorithm called *functional template learning*. This algorithm is competitive with other machine learning techniques in terms of classification accuracy and exhibits several advantages in terms of speed and understandability that make it particularly well suited to interactive, supervised machine learning and autonomous searching of databases.

A prototype version of the learning environment has been implemented as part of a proposed Toolkit for Image Mining. A description of the environment and a preliminary evaluation of performance are presented in this report.

TABLE OF CONTENTS

Abstract	iii
List of Illustrations	vii
1. INTRODUCTION	1
2. THE TOOLKIT ENVIRONMENT	3
3. KNOWLEDGE-BASED IMAGE PROCESSING	7
4. MACHINE LEARNING	11
5. EXAMPLES AND EVALUATION	15
6. DISCUSSION	21
7. FUTURE PLANS	23
8. SUMMARY	25
REFERENCES	27

LIST OF ILLUSTRATIONS

Figure No.		Page
1	User/machine interface.	5
2	Functional template example for thin-line feature detection.	9
3	Construction of scoring function from example and counterexample histograms.	13
4	Learning example.	15
5	Interest image.	17
6	Convergence.	20

1. INTRODUCTION

Vast image databases being accumulated today are overwhelming the capabilities of users to find particular images. For example, much of the data currently obtained from military and civilian sensors are recorded and never accessed because of the search difficulty. And this problem is rapidly getting worse. The National Aeronautics and Space Administration (NASA) is planning a set of earth observing satellites (EOSs) that will generate 100 times the number of images that are currently stored from all remote sensor platforms—more data than are currently stored in the Library of Congress. How these data can be made available to researchers with varying levels of computer expertise and hardware capability is the subject of a continuing national debate [1,2].

This problem is not one that only NASA faces. Military and intelligence communities, medical science, the entertainment industry, the National Oceanic and Atmospheric Administration (NOAA), and law enforcement agencies are all collecting image data at increasingly rapid rates as sensor and mass storage devices become more capable and less expensive. The military and intelligence communities, in particular, employ small armies of analysts to scan image data collected by satellite and airborne sensor platforms. And yet, even in high priority situations, these analysts cannot screen all the data collected in a timely fashion.

Computer assistance will become increasingly necessary to facilitate such searches. Metadata (key words and computed indices used to label each image as a whole), can be used to locate images for some applications. For example, one might ask for Landsat images collected in Florida during June 1993. While very useful, metadata is insufficient and impractical for many search problems, such as finding images of moving targets in military applications, prospecting for minerals in earth satellite data, finding tropical storms in weather satellite images, or attempting to find a face in images generated by an airport surveillance camera. For these and many other applications, the only solution is to search the image contents directly. Note that content-based search and retrieval of images is very different from that for textual libraries. Content-based retrieval of text uses collections of key words, which are part of a finite language. In contrast, image features are near-infinite in variety with no preexisting dictionary and no basis in language or syntax.

The available means for searching through image data on the basis of content are generally primitive. The most primitive are customized algorithms designed to search for some particular signature. While these are often effective, such algorithms are expensive to create and maintain, suitable for only a very limited problem, and their construction and upkeep require an expert in computer vision and image processing.

An active area of recent research and product development has been to build generalized database retrieval or classification tools with user-friendly environments that nonexperts in computer vision could use. Currently, these tools generally allow the user to provide only a single example, be it a drawn shape or a subimage. The algorithm then attempts to characterize the example in some way (e.g., edges, spectral statistics, eigenvalues) and compare these characteristics with those computed for other images or subimages. Differences between example and archive

image characteristics are assimilated into a single “distance” calculation, which is then used as the criterion for selection. While capable of attention-getting demonstrations, these tools have exhibited poor-to-fair performance and have been applied only to small databases in limited domains. Examples of this class of search and classification tools are QBIC, developed by Niblack, et al. at IBM [3], and MultiSpec, developed by Landgrebe and Biehl at Purdue University [4].

The principal failing of this kind of approach is that the user is allowed to present only a single example. As a result, there is no way for the user to know how discriminating the constructed search tool is and to provide feedback to indicate where the algorithm has generated mistakes. There is no way for the search tool to improve with “experience.” Finally, some computed characteristics of an image feature may be inconsistent over multiple images. There is no way for the search tool to discover which characteristics are reliable discriminators. In short, with this approach there is some chance that a user might get good discrimination/classification performance with a single example. But when the user is not getting the desired results, there is no way to make the search tool any better.

Another approach to database search and retrieval is one developed by Turk and Pentland at MIT [5]. Their approach allows the assimilation of example faces into models of facial features called *eigenfaces*. Each face in the database is characterized in terms of differences from these eigenfaces. To recognize a new face, these characterizations are computed and compared with those of faces stored in the database. While eigenfaces is a promising approach to a very hard problem, it still does not promote learning from mistakes or allowing users to shape classification performance to their expectations.

Using tools of knowledge-based signal processing developed at Lincoln Laboratory, a new approach to computer learning, which is the core technology of a proposed Toolkit for Image Mining (TIM), has been designed and implemented. The user and TIM maintain a two-way dialogue, allowing the user to supervise the learning of search tools in a manner analogous to classical conditioning. Learning exhibits rapid convergence to reasonable performance and, when thoroughly trained, these search tools appear to be competitive in discrimination accuracy with other classification techniques such as neural nets and Bayesian classifiers. The learned search tools are simple in structure, self-contained, and can be exported as independent agents to peruse large databases. Moreover, because the search tools are highly modular, it is possible to link them to produce even more complex functionality. This report describes the look and feel of a proposed TIM environment, including details of the implemented learning techniques, and discusses the underlying technology—knowledge-based signal processing—and how it has been used to create a new approach to machine learning. Examples are shown from initial work in spectral pattern recognition and potential applications and extensions of this work are discussed.

2. THE TOOLKIT ENVIRONMENT

The premise of TIM is that a user, having one image containing some object or region, would like to find other such images in a large database. The task is accomplished by a search tool, called an agent, which the user teaches to recognize the discriminating attributes of the feature. Teaching occurs in a supervised learning session in which the user and algorithm continuously provide feedback to each other.

The first step is to fill in information about the data and the nature of the object or region to be sought. This information can include metadata, specifying the name of the sensor and the locations and times of interest. Metadata provides an initial constraint on any searches, reducing the number of images that are actually scanned by the agent. Individual images to be used for training would be specified here as well. Other menus would provide options for those attributes that the algorithm is to consider. For example, if the data source is multispectral, then the value of each spectral band and the value of each ratio of bands might be the considered set of attributes. Various texture and shape analysis techniques might also be considered as candidate attributes.

Following initialization, the main user/machine interface is a screen that looks similar to Figure 1. The screen presents one or more input images (center) and an interest image (right). With the mouse, the user indicates pixels that are examples of the feature to be sought or counterexamples of features to be ignored, represented here as blue (example) and yellow (counterexample) pixels overlaid on one of the input images. Histograms of example and counterexample attribute values (shown on the left) are constructed. Those with the least overlap are determined the best discriminants. An agent is then constructed by exploiting the ranges of values with the least overlap between examples and counterexamples. The agent generates an image in which high interest is assigned to locations that are similar to the set of examples and low interest everywhere else. The user scans the interest image, looking for mismatches between agent output and user expectations. Some pixel location that should have been given a high interest value, but was not, is added by the user to the set of examples. Or some location that was inappropriately given a high interest value is added to the set of counterexamples. With each new input, the algorithm revises the “model” of user intentions, produces a new interest image, and the process repeats until the user is satisfied with agent performance on a single input image.

Once satisfied, the user must select some criterion for image retrieval. For example, the user might request that the agent retrieve any image for which the average interest value is greater than some threshold t . Or perhaps the agent should retrieve any image in which the number of interest values greater than t is more than 10% of the total number of pixels in the image.

The agent is then ready to embark on a search. While it could certainly search a database resident on the training workstation, the main goal is to create an agent that can enter a super-computer serving a large database, for example, one of the Data Archive and Analysis Centers (DAACs) that NASA has established as part of the EOS Data and Information System (EOSDIS). One simple strategy would be for the agent code and parameters to be e-mailed to the DAAC host,

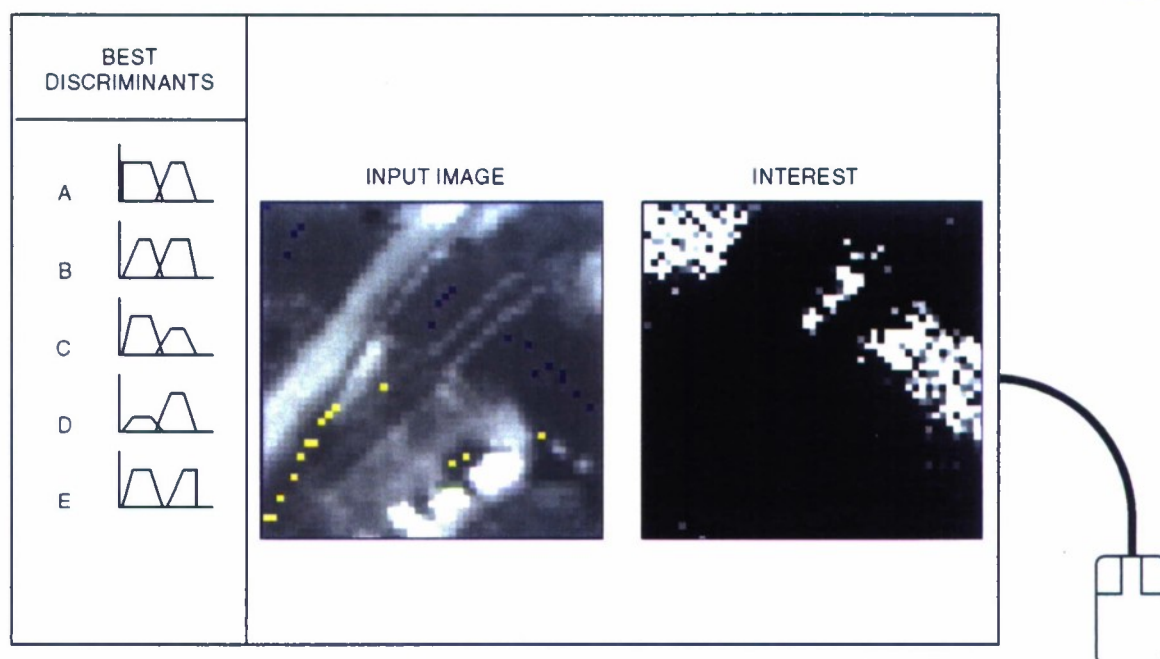


Figure 1. User/machine interface. Notional appearance of the interactive learning environment in TIM.

which would know by some prearranged code that it should compile, load, and start the agent. Metadata collected during agent initialization would be used to constrain the search. Time limits could be set to protect both the user and the user community from runaway, forgotten, or poorly learned agents. Perhaps the agent might be configured either to search through archived data or monitor the stream of recently processed images on their way to archival storage.

The user explores the retrieved input images and computed interest images both to assess agent performance and to look for interesting object or region variants found by the agent. At this point, the user has the option of reentering the learning environment to further refine the agent.

The interest image also provides a limited means for quantitative analysis. Scientists or policy makers interested in spatial measurements or in trend analysis would be able to quantify the extent of high interest regions with various statistical characterizations and perhaps look for changes over time.

3. KNOWLEDGE-BASED IMAGE PROCESSING

The core technology of TIM is a new machine learning algorithm, based on techniques of knowledge-based image processing developed at Lincoln Laboratory. More traditional approaches to machine learning, such as neural nets, might also be used. However, as described later, the new approach has several advantages over other machine learning approaches and is particularly well suited to the TIM environment.

Techniques of knowledge-based image processing have been developed at Lincoln Laboratory in the context of automatic target recognition [6]. These techniques have been successfully extended to other areas, including the automatic detection of hazardous weather in Doppler radar images [7,8,9]. Several weather detection algorithms constructed by means of this approach are being operationally tested at several airports across the country. These new algorithms have consistently demonstrated capabilities that are much improved over the prior state of the art. Knowledge-based image processing is based on two particular tools: interest images and functional template correlation.

An interest image is a spatial map of evidence for the presence of some feature [10]. To be useful in the context of a search, that feature should be selectively indicative of some region or object that is being sought. Pixel values range from [0,1] with higher pixel values reflecting greater confidence that an intended feature is present at that location. Interest is particularly effective as a “common denominator” for data fusion. Input data from a variety of sensor sources can be transformed into one or more interest images highlighting some set of features thought to be indicative of a target. With simple or arbitrarily complex rules of arithmetic, fuzzy logic, or statistics, these individual interest images can be assimilated into a single combined interest image. Knowledge of circumstances under which each interest image is a good discriminant can be used to guide the weighting of each individual interest image during fusion. Ultimately, a detection algorithm with very complex properties can be assembled from a set of simple feature detectors, each of which generates an interest image. In addition to their role in data fusion, interest images provide a means of representing information for human interpretation. Highlighting the features of interest cues human attention to a particular feature that would be difficult to spot in raw image data.

Functional template correlation (FTC) is a generalized matched filter used to create customized, knowledge-based image processing operations [11,12]. FTC, originally developed as a means of computing interest images, incorporates aspects of fuzzy set theory to transform raw data into maps of match scores in the range [0,1]. FTC can most easily be understood as a simple extension of *autocorrelation*. Given some input image I , an output image O is generated in autocorrelation by matching a kernel K (essentially a small image of image values) at each pixel location I_{xy} . Each element of K is a literal image value expected to be present for a good match. When K is tested at each pixel location I_{xy} , elements of K overlay pixels in the local neighborhood of I_{xy} . Each element of K and the corresponding superimposed pixel value are multiplied together.

The sum of these products is the match score placed in O_{xy} . If the shape to be matched can vary in orientation, then the pixel I_{xy} is probed by kernel K at multiple orientations. The score assigned to O_{xy} is the maximum across all orientations.

FTC is fundamentally the same operation with one important exception—where the kernel of autocorrelation consists of literal image values (essentially a subimage of the image to be probed) the kernel used in FTC is a set of indices, each referencing a scoring function. These scoring functions encode a mapping between input image values and scores to be returned. High scores are returned whenever the input image value falls within the fuzzy limits of expected values. Low scores are returned whenever the input value falls outside these limits. The set of scores returned for each element of the kernel K are averaged and clipped to the continuous range $[0,1]$. A *functional template*, then, is a kernel of indices and their associated scoring functions. As in autocorrelation, if the feature being sought can vary in orientation, then the match score is the maximum average score computed across multiple orientations of the kernel.

As an example, consider the functional template shown in Figure 2 that is designed to detect gust fronts in radar reflectivity data. Gust fronts are observed as thin lines of moderate reflectivity (approximately 0 to 20 dBZ) that are flanked on both sides by low reflectivity values (approximately -10 to 0 dBZ). On the left is the template kernel consisting of integers, corresponding to the two scoring functions shown on the right. Elements of the kernel that do not have an index form guard regions in which image values are ignored and have no effect on match scores. Scoring function 0, corresponding to the flanking regions of low reflectivity, returns a maximal score of 1 for image values in the interval of -20 to -5 dBZ, a gradually decreasing score for image values in the interval -5 to 10 dBZ, and a score of -2 for image values larger than 10 dBZ. Scoring function 1, corresponding to the center of the kernel where moderate reflectivity values are expected, returns maximal scores in the interval between 5 and 12.5 dBZ with gradually decreasing scores for both higher and lower image values.

In general, by increasing or decreasing the interval over which affirming scores (i.e., > 0.5) are returned, scoring functions can encode varying degrees of uncertainty with regard to allowable image values. In addition, knowledge of how a feature or object appears in sensor imagery can be encoded in scoring functions. With various design strategies, the interfering effects of occlusion, distortion, noise, and clutter can be minimized [11]. As a consequence, functional templates customized for specific applications are generally more robust than standard generic signal processing operations. In the thin-line matched filter example shown in Figure 2, the filter does not simply find thin lines, but can select those having reflectivity values within a particular range. FTC has been used as a direct, one-step means of detecting 3-D objects and can be used for implementing fuzzy, knowledge-based versions of edge detection, thin-line filtering, thin-line smoothing, shape matching, skeletonizing, and eroding.

A means of data fusion has been implemented in FTC that allows multiple functional templates to probe input images in tandem, producing a single output interest image. This technique assumes that each input image is pixel-registered to the others and that the functional templates

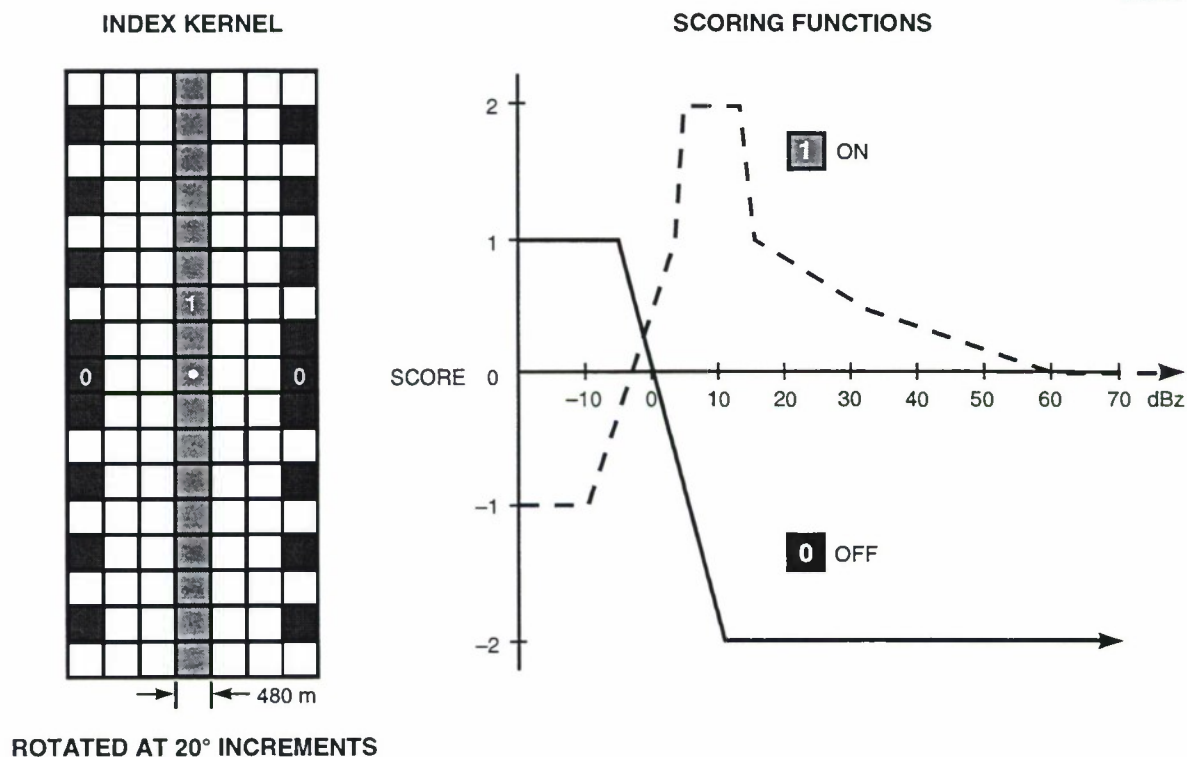


Figure 2. Functional template example for thin-line feature detection.

all share a common center of rotation. A match score for pixel O_{xy} is computed as an average of the set of scores computed for all elements of all kernels. Each template or scoring function can be assigned a unique weight in the range $[0,1]$, allowing knowledge of the reliability of each template or function to bias the average. If the target being sought can vary in orientation, then the kernels of all functional templates are rotated together in lock step.

4. MACHINE LEARNING

To probe multispectral images, functional templates are constructed for each spectral band. Each functional template consists of a kernel having a single element, along with a single scoring function. Transformations of multiple bands into a single value, such as the ratio of two bands, might also be considered as inputs and likewise have a functional template. In addition, other transformations of the input data might use a window operation to compute for each pixel an index that is indicative of the spatial properties or texture in the neighborhood surrounding the pixel. Most useful are transformations that compute various attributes of texture. For example, local neighborhood variance can be computed for each pixel location in an image. Other transformations might include fractal dimension, contrast, periodicity, orientation, and parallelism. The functional templates for each spectral band, spectral transformations, and textural transformations would be applied in tandem, producing a single weighted average (i.e., interest value) for each pixel of the output interest image.

Given this structure for spectral and textural functional template correlation, functional template learning is implemented by constructing scoring functions and computing weights from a set of examples and counterexamples. Briefly, the current approach is based on the construction of distribution histograms, one for example and another for counterexample values of each attribute (spectral band or transformation). The weight for a functional template is computed from a measure of the overall extent of overlap between the example and counterexample distributions. The weight is 1 if there is no overlap between examples and counterexamples. The weight is 0 when the distributions of examples and counterexamples are the same. The scoring function is a mapping constructed from a comparison of the extent of overlap at each input image value. For example, high scores are returned for input values that have many examples and few or no counterexamples. Similarly, low scores are returned for values that have many counterexamples and few or no examples. What follows is a more detailed description of how a single scoring function and weight is computed for an attribute.

The mean and standard deviation of the set of examples (not counterexamples) are computed for some attribute. These statistics are then used to scale image values to a range of 0 to 255, such that the mean value is mapped to 128, while 0 and 255 are mapped to the input values that are 3 standard deviations below and above the mean, respectively. Input values more than 3 standard deviations from the mean are clipped to 0 and 255. This scaling is based on the example statistics and applied to all example and counterexample attribute values.

A histogram is then constructed from the set of scaled example values. Each value is added to the histogram by incrementing the count of the corresponding bin. For instance, if the scaled value is 201, then the histogram value of bin 201 is incremented by 1. If the total number of examples is small, then some number of bins to either side are also incremented to approximate a fully populated distribution. The number of neighboring bins to be incremented is currently computed as follows:

$$width = INT(p \times (1.0 - \frac{n}{256}))$$

where n is the number of example and counterexample values, p is a user-tunable parameter (typically set to 8), and $INT()$ is the truncation operator. A counterexample histogram is likewise constructed from the set of scaled counterexample values.

Each functional template weight is computed as follows:

$$weight = 2 \times \left(0.5 - \frac{\sum_{i=0}^{255} \text{MIN}(H_i^{ex}, H_i^{cex})}{\sum_{i=0}^{255} (H_i^{ex} + H_i^{cex})} \right)$$

where i is the bin number and H^{ex} and H^{cex} are the example and counterexample histograms. The attributes are ranked by weight in decreasing order. Scoring functions are then constructed for the m highest weighted attributes, called the *best discriminants*. The parameter m is user tunable.

Scoring functions are computed from a comparison of the corresponding example and counterexample histograms as shown in Figure 3. Each scoring function is implemented as a lookup table of 256 scores, one for each possible scaled image value. The score F_i to be returned for each input image value i is computed as follows:

$$F_i = \begin{cases} 0.0 & \text{if } H_i^{ex} = 0 \text{ and } H_i^{cex} = 0 \\ 16 \times R(H_i^{cex}, \sum_{i=0}^{255} H_i^{cex}, 0.0, 0.25) & \text{if } H_i^{ex} = 0 \\ 16 \times R(H_i^{ex}, \sum_{i=0}^{255} H_i^{ex}, 0.0, 0.25) & \text{if } H_i^{cex} = 0 \\ 16 \times R(H_i^{ex}, H_i^{cex}, 1.0, 40.0) & \text{if } H_i^{cex} > H_i^{ex} \\ 16 \times R(H_i^{cex}, H_i^{ex}, 1.0, 40.0) & \text{otherwise} \end{cases}$$

where $R(a, b, c, d)$ is the ramp function

$$R(a, b, c, d) = \text{MIN} \left(1.0, \frac{\text{MAX}(0.0, \frac{a}{b} - c)}{d - c} \right).$$

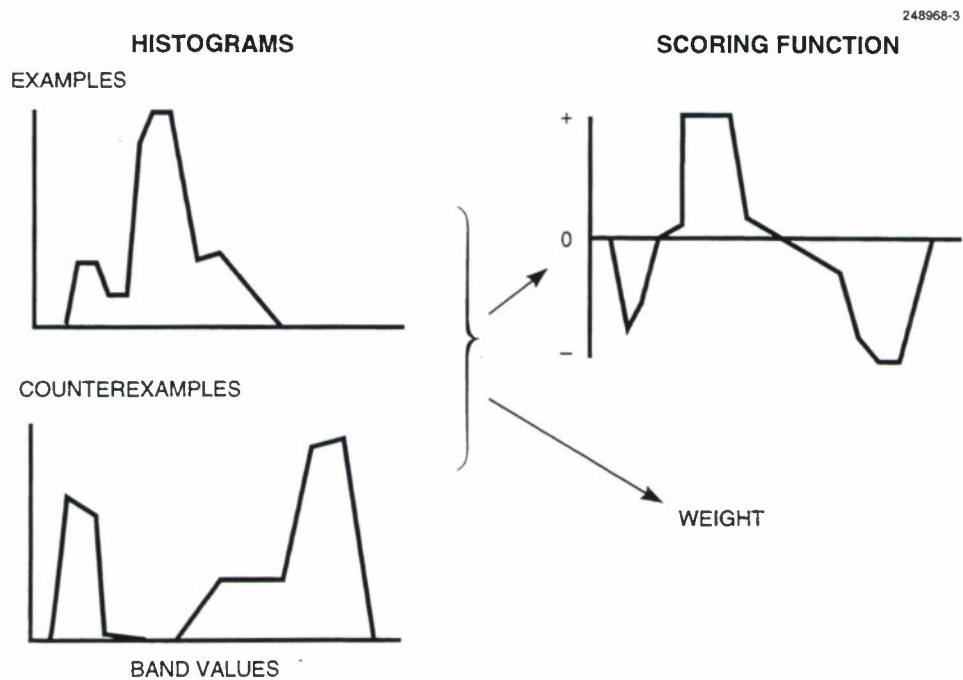


Figure 3. Construction of scoring function from example and counterexample histograms.

5. EXAMPLES AND EVALUATION

Agent training and use are illustrated with two different examples. The first uses three-band (long wave, medium wave, and short wave) infrared images of urban and suburban scenes. Images were divided into 50×50 pixel tiles. Figure 4 shows one such image of a town in Michigan and the three spectral bands of a tile (tile 40 in the full image) showing two bridges crossing a river. User intent is to create an agent that will identify water. The blue and yellow pixels in the frame marked LW indicate the examples and counterexamples selected by the user for training. On the right is the interest image produced by the agent, which is selectively highlighting the river.

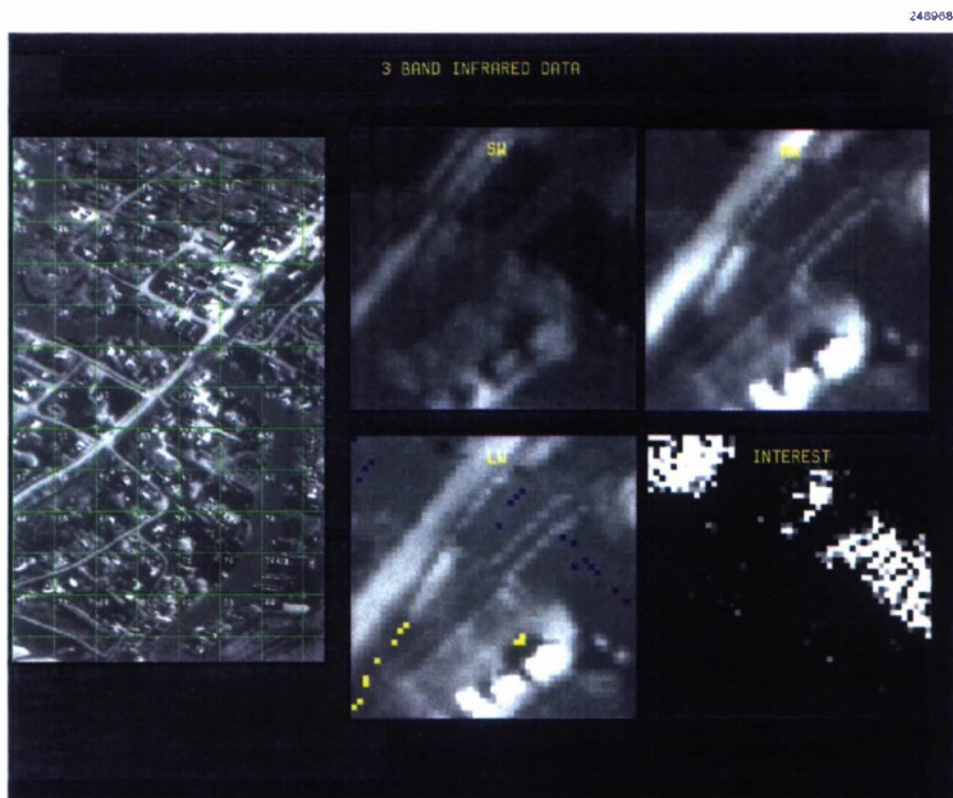


Figure 4. Example set of examples and counterexamples and the resulting interest image, highlighting a river in three-band infrared images.

To search through a database of images for more images of water, the agent must be assigned some selection criterion. This criterion is a threshold applied to some statistic of interest values. One possible selection threshold would be to take those images that have average interest images

greater than 0.1. Another alternative would be to select images in which 10% or more of interest pixels have values above 0.5. Figure 5 illustrates the selection process. On the left is the full size image shown in Figure 4. On the right is the interest image generated by applying to the whole image the agent taught to recognize water. The yellow boxes enclose tiles that have average interest values of 0.06 (out of the range $[0,1]$) or higher. Search functionality is demonstrated by looking for high interest tiles within a large image. The same mechanism would apply to the task of looking for high interest images within a large database.

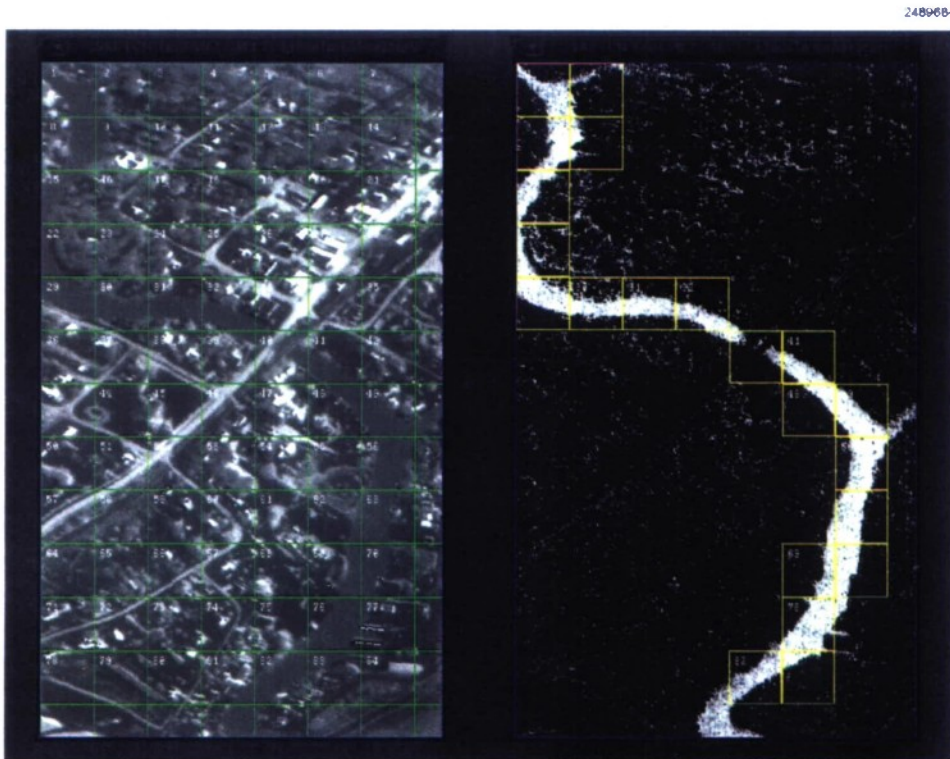


Figure 5. Interest image and selected subimages resulting from the application of the functional template constructed in Figure 4.

A more difficult task is the classification of vegetation types using Landsat Thematic Mapper (TM) images. Huber and Casler [13] have described work in which they evaluated a variety of simple classification metrics and formulas developed by wildlife biologists. Their area of study, a region in central Colorado, was classified pixel by pixel into 14 vegetation types. Actual ground data were collected and used for scoring. They concluded that “using remote sensing information

in complex terrains can be a difficult and relatively inaccurate process.” With no technique did correct classifications exceed 50% with high numbers of false alarms. When digital elevation model data were used to determine the direction in which terrain was facing, they reached nearly 70% detection, although still with substantial numbers of false alarms.

Using the same TM data, Augusteijn et al. [14] used a variety of texture analysis techniques in combination with a cascade-correlation neural net (an enhanced version of back propagation) [15] to classify homogeneous regions. The data chosen by them for training and testing consisted of 8×8 pixel boxes containing only 1 vegetation type. Of the 14 vegetation types, only 9 were available in boxes of this size. The task was to train the neural net on half these boxes and then attempt to identify vegetation types in the remaining boxes. Using this approach, they were able to achieve approximately 99% correct classifications.

This same TM image was used here to perform a preliminary evaluation of functional template learning. Using the truth image to guide training in a local area, agents were trained to identify each vegetation type. A 100×100 pixel image was used for training. Convergence to a solution was typically rapid. For example, Figure 6 shows (for one of the agents) the number of correct detections and false alarms as each example or counterexample was added. Even after only 50 inputs, the agent was correctly classifying 80% of all pixels having this vegetation type with fewer than 5% false alarms. The average classification accuracy for 13 trained agents (1 vegetation type was not present in the training image) was 94% correct detections with 6% false alarms. These agents were tested on a neighboring 100×100 subimage, achieving 86% correct detections and 13% false alarms.

These results are clearly better than those reported by Huber and Casler, even when they incorporated topographic data in addition to the six TM bands. It is more difficult to compare current results with those presented by Augusteijn et al. Both studies used six-band, single-pixel TM data. However, where the current study attempted to classify every pixel, the previous study attempted to classify homogenous 8×8 patches. Many, if not most, of the classification mistakes made by the system reported here were near the boundaries between regions. These boundaries are likely to have transitional spectral characteristics, which should be more ambiguous. By only considering large patches, the previous study avoided any attempt to classify these boundary zones.

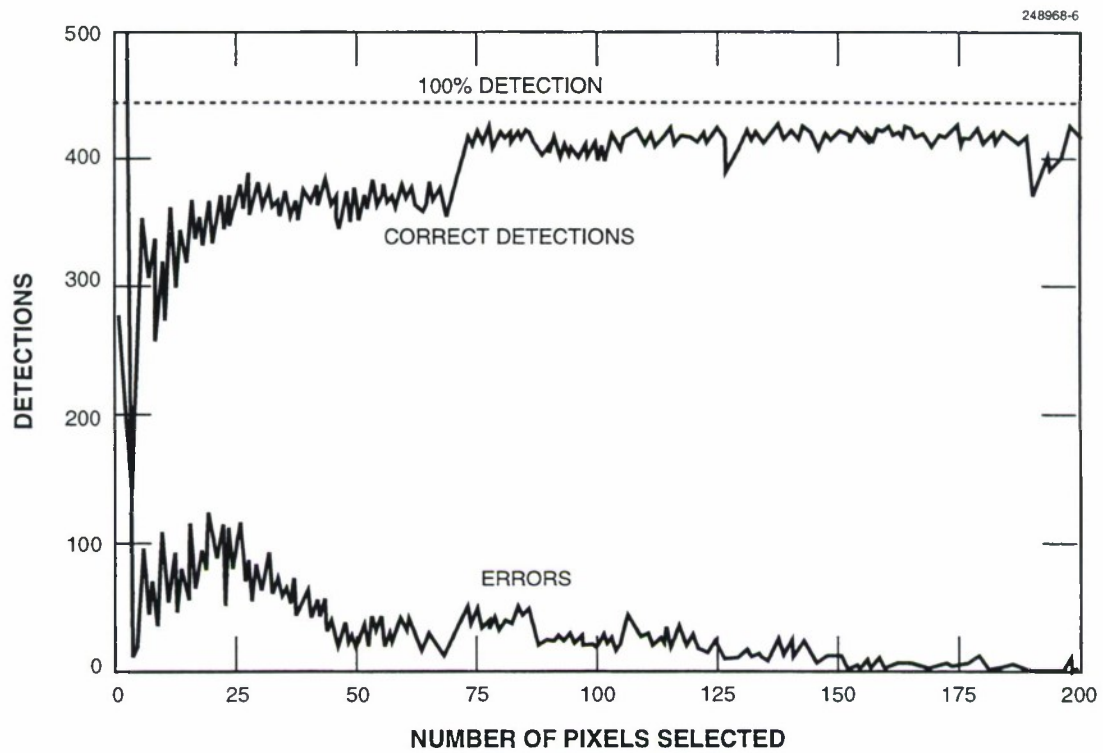


Figure 6. The number of correct detections and number of errors for a single agent (functional template) plotted for each successive input example or counterexample.

6. DISCUSSION

Databases will soon contain far more data than humans can peruse in any useful interval of time. Other limitations, such as communications bandwidth and computer resources, will impose costs for each image retrieved. However, the data handling capabilities of computers and communications networks are increasing far more rapidly than the abilities of humans. Ultimately, the human limitations must be addressed.

The current design enables a user and a machine learning algorithm to collaborate in the training of an agent to recognize some particular spectral or textural pattern. Provided with examples of the intended pattern and counterexamples of patterns to ignore, the learning algorithm attempts to create a model of the intended pattern. The trained agent would then be sent into a large database to autonomously search for other images containing the same pattern. As mistakes are made, the user can continue to refine the agent by adding more examples and counterexamples.

Several content-based image retrieval systems already exist in which the user provides an example of what is wanted. However, in all previous systems the user can supply only a single example: there is no capability for learning from mistakes. In single example systems, a number of computed feature attributes are indiscriminately matched using a Euclidean distance metric against those computed for database images. But not every attribute is discriminating. Unlike the various single example mechanisms, FTC can learn the attributes that are consistently discriminating over a variety of examples.

An application that illustrates the advantages of learning from mistakes is that of prospecting. Suppose geologists find an unusual mineral deposit. Using hyperspectral image data, they might indicate the location of the deposit in TIM. From the single example the constructed agent would create an interest image highlighting the known deposit and any other locations having similar characteristics. The geologists might then visit some of these other locations, either confirming or denying agent assessment. As feedback is given in the form of new examples or counterexamples, the best discriminants would be discovered.

Most learning algorithms are set up to be trained from a selection of inputs that are arbitrarily fixed in order of presentation prior to learning. Selection of a next input is unrelated to how well the discrimination task is learned. In contrast, the learning environment in TIM promotes a more directed selection of inputs. The use of interest images as a means of visual feedback allows the user to select those inputs that will most rapidly correct errors in discrimination. Consequently, if a solution is possible, then convergence to a solution is rapid.

This collaborative approach to pattern learning incorporated in TIM could be accomplished with other machine learning techniques. Functional template learning, however, has many advantages over other techniques that make it well suited to the TIM environment. For example, neural nets have become the standard against which machine learning techniques are compared. They often achieve good classification performance and are adaptable to a wide variety of problems.

However, neural nets tend to exhibit a number of awkward disadvantages that are not apparent with functional template learning.

- A large number of inputs are required before neural nets converge on a solution. As currently implemented, functional template learning is capable of producing reasonably accurate agents in as little as 5 to 10 inputs.
- It is difficult to uncover the attributes used by a neural net to discriminate between classes. Functional template learning easily interprets the ranking of weights and the appearance of scoring functions, which are easily edited to produce predictable performance changes.
- The classification behavior of a trained neural net often varies with the order of presentation of training data. Because the weights and scoring functions of functional templates are based on the values that are currently in the example and counterexample sets, they are necessarily the same, regardless of order of input.
- In search mode, trained neural nets are more computationally expensive than functional templates. Each neuron of a neural net receives an input from each neuron in the preceding level and sends an output to each neuron in the succeeding level. The combinatoric pattern of calculations is not present in FTC. Moreover, FTC works using table lookups, which is much faster than the calculations needed at each neuron of a neural net.

The goal of TIM is not necessarily to provide perfect discrimination; functional template learning does not have to be as good a classifier as neural nets to be useful in the context of database search. In a search for some particular event or feature, TIM can provide a prescreening service to reduce the number of images that users need to download and observe.

TIM would allow a user with limited computer experience to build useful computer vision tools and encourage users to explore image databases. Among other things, users might

- cover more imagery than previously would have been possible, given limitations in human endurance and attention;
- discover better formulas for computing indices for such image features as vegetation and clouds; and
- discover relationships between regions or objects that had not been noticed before.

Such discoveries have always been possible; however, cost has been a limiting factor because such exploration has typically been accomplished by customized programs developed by people with both computer and specific application expertise. With TIM, a larger community of users with application experience but limited computer experience can inexpensively do the explorations for themselves.

7. FUTURE PLANS

The primary goal is to provide a full working TIM environment that is capable of interacting with a large database such as the DAACs being assembled under the EOSDIS project. One objective is to work out strategies to prescreen data using metadata; others include the rules and handshaking that allow an agent to be exported to a DAAC supercomputer, compiled, started, and terminated. Because several research organizations are working on DAAC interfaces, collaboration might be useful.

A secondary goal is extending the machine learning techniques to shape. The current implementation of functional template learning is capable of learning patterns based on point characteristics such as spectrum, polarization, and textural transformations. FTC was originally constructed as a shape-matching tool, meaning that the same FTC engine that would be packaged as part of any agent already has the capability to search for shapes; missing is an interactive means to learn them, where the learning algorithm produces an interest image indicating locations of matches and a means for the user to indicate mistakes. To do so, a means of registering each example to a common reference frame must be developed. For example, if a particular shape varies in orientation, then each example of that shape must be rotated to a common orientation. Similarly, the centers and scaling of each instance of a shape must be registered.

Interest provides a “common denominator” for data fusion in detection tasks. Any two interest images can be combined using simple rules of pixel-level arithmetic or fuzzy logic. Using this technique, computer vision systems have been constructed with complicated rules of behavior from simple functional templates. Similarly, agents (collections of functional templates) learned in the TIM environment might be assembled into more complicated algorithms. One simple example is using a set of agents, each with a different resolution. A coarse resolution agent, presumably faster but more prone to errors than a fine resolution agent, might prescreen a large set of data. Those images selected by the coarse resolution agent would then be searched by the fine resolution agent. Another way of combining the outputs of agents is to collect the outputs of several agents, each looking for some different feature of a region or object. A weighted average of these interest images might better highlight the locations of some region or object than any individual interest image. Functional templates can also take other interest images as input, allowing sequential processing. For example, the interest image shown in Figure 5 might be improved by applying a circular mean filter, which would tend to suppress any isolated high interest values and raise any isolated low interest values.

A final area of expansion is the use of principal component analysis (PCA) and related techniques to explore the assembled sets of example and counterexample values. The motivation of PCA is to find ways of mathematically combining groups of attributes into a new single attribute that is a better class discriminant than any of the original, individual attributes. A ratio is one simple example of how two attributes can be combined mathematically into one. Such new attributes could then be considered in addition to the originals used for functional template learning.

If these new attributes prove to be better discriminants, then their weights are higher and tend to dominant the characteristics of the agent.

8. SUMMARY

The ideas presented here would produce a TIM workstation environment that would assist users in creating agents that could recognize some spectral or textural pattern. Users having no knowledge of computer vision could teach these agents by indicating locations where the agent misclassifies patterns. Once trained, these agents could search large image databases for matching patterns. Alternatively, they might be incorporated into more complicated computer vision algorithms for planning policy, prospecting, or doing research to understand the similarities and differences between regions or objects.

Agent training is done during interactive sessions in which the user and algorithm provide constant feedback to each other. The underlying mechanism for agent training is based on a new technique called functional template learning. While competitive with neural nets in classification performance, this technique has several advantages that makes it particularly suited to the interactive mode of agent training, including rapid convergence, input order independence, and easily understanding the attributes used for discrimination. In addition, agents based on functional template correlation are, relatively speaking, computationally efficient search tools.

Using these techniques, a prototype learning environment that would be included in a fully developed TIM has been implemented and is being tested.

REFERENCES

1. "NASA's EOSDIS development approach is risky," U.S. General Accounting Office Rep. GAO/IMTEC-92-94 (February 1992).
2. "Broader involvement of the EOSDIS community is needed," U.S. General Accounting Office Rep. GAO/IMTEC-92-40 (May 1992).
3. W. Niblack, R. Barber, W. Equitz, M. Flickner, E. Glasman, D. Petkovic, P. Yanker, C. Faloutsos, and G. Taubin, "The QBIC project: Querying images by content using color, texture, and shape," *Computer Science* (February 1993).
4. D. Landgrebe and L. Biehl, *An Introduction to MultiSpec*, School of Electrical Engineering, Purdue University, 2.15.94 ed (February 1994).
5. M. Turk and A. Pentland, "Eigenfaces for recognition," *J. Cognitive Neuroscience* **3**, 71-86 (1991).
6. R.L. Delanoy, J.G. Verly, and D.E. Dudgeon, "Machine intelligent automatic recognition of critical mobile targets in laser radar imagery," *Linc. Lab. J.* **6**, 161-186 (Spring 1993).
7. R.L. Delanoy and S.W. Troxel, "Automated gust front detection using knowledge-based signal processing," *IEEE 1993 Natl. Radar Conf. Radar Meteorology*, Norman, Okla. (May 1993).
8. R.L. Delanoy, J.G. Verly, and D.E. Dudgeon, "Pixel-level fusion using interest images," Technical Rep. 979, Lexington, Mass.: MIT Lincoln Laboratory (May 1993). DTIC AD-A-2666400.
9. R.L. Delanoy, J.G. Verly, and D.E. Dudgeon, "Functional templates and their application to 3-D object recognition," *Proc. Int. Conf. Acoustics, Speech, Sig. Proc.*, San Francisco, Calif. (March 1992).
10. R.L. Delanoy and J.G. Verly, Computer apparatus and method for fuzzy template matching using a scoring function, U.S. Patent No. 5,222,155 (June 1993).
11. H.P. Huber and K.E. Casler, "Initial analysis of Landsat TM data for elk habitat mapping," *Int. J. Remote Sensing* **11**, 907-912 (1990).
12. M.F. Augusteijn, L.W. Clemens, and K.A. Shaw, "Performance evaluation of texture measurements for ground cover identification in satellite images by means of a neural network classifier," Technical Rep. EAS-CS-93-8, University of Colorado, Colorado Springs (October 1993).
13. S.E. Fahlman and C. Lebiere, "The cascade-correlation learning architecture," *Advances in Neural Information Processing Systems 2*, D. Touretzky, ed., San Mateo, Calif.: Morgan Kaufmann (1990).

REPORT DOCUMENTATION PAGE			Form Approved OMB No. 0704-0188	
<small>Public reporting burden for this collection of information is estimated to average 1 hour per response, including the time for reviewing instructions, searching existing data sources, gathering and maintaining the data needed, and completing and reviewing the collection of information. Send comments regarding this burden estimate or any other aspect of this collection of information, including suggestions for reducing this burden, to Washington Headquarters Services, Directorate for Information Operations and Reports, 1215 Jefferson Davis Highway, Suite 1204, Arlington, VA 22202-4302, and to the Office of Management and Budget, Paperwork Reduction Project (0704-0188), Washington, DC 20503.</small>				
1. AGENCY USE ONLY (Leave blank)	2. REPORT DATE 6 March 1995	3. REPORT TYPE AND DATES COVERED Technical Report		
4. TITLE AND SUBTITLE Machine Learning for a Toolkit for Image Mining		5. FUNDING NUMBERS C — F19628-95-C-0002 PR — 1		
6. AUTHOR(S) Richard L. Delanoy and Richard J. Sasiela				
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) Lincoln Laboratory, MIT P.O. Box 73 Lexington, MA 02173-9108		8. PERFORMING ORGANIZATION REPORT NUMBER TR-1017		
9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES) Wright Laboratory/AARA Building 23 2010 5th Street Wright-Patterson AFB, OH 45433-7001		10. SPONSORING/MONITORING AGENCY REPORT NUMBER ESC-TR-94-112		
11. SUPPLEMENTARY NOTES None				
12a. DISTRIBUTION/AVAILABILITY STATEMENT Approved for public release; distribution is unlimited.		12b. DISTRIBUTION CODE		
13. ABSTRACT (Maximum 200 words) <p>Given the exponential growth in the number of images stored in databases around the world, there is a clear need for computer-assisted means of searching images. Keywords and computed indices, referred to as "metadata," can be used to tag images for subsequent retrieval. Most potential applications, however, will require searching for images on the basis of image content, but existing techniques for content-based search are either too slow, customized at great expense for a single application, or simply do not work well.</p> <p>An environment has been created enabling a user with limited computer skills to interactively train a computer algorithm to recognize patterns of spectral or textural features in imagery. The trained algorithm can then be exported as an independent agent to search large databases for matching image patterns. Following their retrieval, a user can further refine agent performance by indicating mistakes. The final product is a search tool capable of prescreening and highlighting images, significantly reducing the workload of analysis attempting to detect regions or objects in imagery.</p> <p>An approach to interactive learning has been developed as part of this environment. Based on techniques of knowledge-based image processing, this approach uses interest images to provide a means of continuous feedback of algorithm performance to the user, who in turn responds by indicating where the algorithm is making mistakes. The set of user-indicated examples and counterexamples form the inputs to a new machine learning algorithm called <i>functional template learning</i>. This algorithm is competitive with other machine learning techniques in terms of classification accuracy and exhibits several advantages in terms of speed and understandability that make it particularly well suited to interactive, supervised machine learning and autonomous searching of databases.</p> <p>A prototype version of the learning environment has been implemented as part of a proposed Toolkit for Image Mining. A description of the environment and a preliminary evaluation of performance are presented in this report.</p>				
14. SUBJECT TERMS contents-based image retrieval remote sensing functional template correlation machine learning		15. NUMBER OF PAGES 38		
		16. PRICE CODE		
17. SECURITY CLASSIFICATION OF REPORT Unclassified	18. SECURITY CLASSIFICATION OF THIS PAGE Unclassified	19. SECURITY CLASSIFICATION OF ABSTRACT Unclassified	20. LIMITATION OF ABSTRACT Same as Report	